- Build-in functions
  - Map(function_name , list_name)

```
def names(text):
    return f"- {text} -"

words = ["amira" , "aya" , "yoka" , "meroo"]
words_map = map(names , words)
for name in words:
    print(name)
    # - amira -
    # - aya -
    # - yoka -
    # - meroo -
```

  - filter()

```
def number_filter(num):
    if num > 10:
        return num
numbers = [10 , 20, 3, 50 , 11 , 5]
list_filter = filter(number_filter , numbers)
for num in list_filter:
    print(f"{num}" , end = " ") #20 50 11
```

```
def number_filter(num):
    if num == 0:
        return True
numbers = [10 , 0, 3, 0 , 11 , 5]
list_filter = filter(number_filter , numbers)
for num in list_filter:
    print(f"{num}" , end = " ") # 0 0
```

  - reduce()

```
from functools import reduce
def sumAll(num1 , num2):
    return num1 + num2
list_reduce = [1 , 5, 10 , 10 , 20, 15] #61
    # (((((1+5) + 10) + 20 ) + 15)
result = reduce(sumAll , list_reduce)
print(result)
```

  - enumerate(iterator , start)

```
mySkills = ["Html" , "Css" , "JS" , "C" , "C++"]
mySkillsCounter = enumerate(mySkills , 10)
for counter in mySkillsCounter:
    print(counter)
# (10, 'Html')
# (11, 'Css')
# (12, 'JS')
# (13, 'C')
# (14, 'C++')
```

  - Print()
  - Reversed()
  - Id(variable) => place of variable in memory

- Type()
- Help("keywords")
- Reshape(metrics , (height , width)
- Dir(module_name)
- All(list_name)
- Any(list_name)
- Bin(number) => convert decimal to binary
- Sum(alterable , start)
- Round(number , num_of_digits)
- Range(start , end , step)
  - Range(10) => from 0 to 10
- Sep = " "

```
print("Hello" , "meroo" , "how" , "old" , "are you" , sep = " & ")
# Hello & meroo & how & old & are you
```

- end = "\n"

```
print("amira" , end = " ")
print("hassan") #amira hassan
print("sobhi")#sobhi
```

- abs()
- pow(base , exp , mod)
- min()

❖ Variables

- X -> without datatype
- Global x
  - x = 5
  - def num():
  - global x
  - x = 14
  - num()
  - print(x) #14
- Var_name.isidentifier() #check on variable name if it valid or no

```
a = "aa--nn"
print(a.isidentifier()) # false
```

- Var_name.isalpha()

```
a = "21432ddvdx"
print(a.isalpha()) # false
```

- Var_name.isalnum()

```
a = "fdg5546cs*&cdsvv"
print(a.isalnum()) # false
```

❖ Data types

- Complex

```
complexNumber = 3 + 5j
print(type(complexNumber)) #<class 'complex'>
```

```
print("realNumber {}" . format(complexNumber.real))
#realNumber 3.0
print("imagNumber {}" . format(complexNumber.imag))
#imagNumber 5.0
```

- Numbers
  - Int
  - Float
  - complex
- Strings
  - “”” -> to print more than one line
  - Methods
    - Len(str)
    - Str.upper()
    - Str.lower()
    - Str.islower()
    - Str.isspace()
    - Str.expandtabs(2)
    - Str.replace(“old_word” , “new_word” , limit if you want)
    -
    - Split Text

```
str = "i love my name"
print(str.split()) # ['i', 'love', 'my',
'name']
print(str.split( " ", 2)) # ['i', 'love', 'my
name']
print(str.rsplit(" " , 2)) # ['i love', 'my',
'name']
str = """first
second third
"""
print(str.splitlines()) # ['first', 'second
third']
```

    - put the text in the middle

```
str = "amira"
print(str.center(10)) #   amira
print(str.center(10,"#")) # ##amira###
```

    - Str.count(“word that you wanna search it” , start , end)
    - Str.swapcase() #replce lowercase with uppercase and vice versa
    - Search
      - Str.startwith(“u”,start , end)
      - Str.endwith(“u”,start , end)
      - Str.find(“u” , start , end) # if u not found it will return -1
      - Str.index(“u” , start , end) # if u not found it will return error
    - Put spaces || characters

```
str = "Meroo"
print(str.rjust(10 , "*")) #*****Meroo
print(str.ljust(10 , "*")) #Meroo*****
```

- Remove spaces

```
str = "   mer  ooo   "
```

  - Str.strip() # mer ooo.
  - Str.lstrip() # mer ooo   .
  - Str.rstrip() #   mer ooo.

- Remove characters

```
str = "***mer**ooo***"
```

  - Str.strip("*") #mer**ooo
  - Str.lstrip("*") #mer**ooo***
  - Str.rstrip("*") #***mer**ooo

- Str.title()
- Str.istitle()

```
str = "info 3d cycle remove 2am"
```

  - # Info 3D Cycle Remove 2Am

- Str.capitalize()
  - # Info 3d cycle remove 2am
- Str.zfill(length of num)

```
a , b , c, d = "1" , "12" , "123" , "1234"
print(a.zfill(3)) #001
print(b.zfill(3)) #012
print(c.zfill(3)) #123
print(d.zfill(3)) #1234
```

-

- Boolean
  - Methods
    - Bool(value)
- Lists (mutable)
  - Writing method
    - numbers = [3 , 5, 9,7]
      - print(numbers[1:3]) [5,9]
      - print(numbers[1]) 5
      - print(numbers[-1]) 7
      - print(numbers[-2::]) [9,7]
      - print(numbers[::3]) [3,7]
    - numbers = [3 , 5, [1 , 2], 7 , 8]
  - functions
    - .append(value) to add a value to the end of the list
    - .insert(index,value you want it)
    - .extend([value,value,…]) to add more than one value
    - Sum(list_name)
    - L1 = sorted(list_name)

- .sort()
- .reverse()
- ‘’.join(list_name) to concatenate the list
- .remove(item)
- Del list_name[1]
- .pop() to return last element and use it if I want
- .pop(2) 2 -> the index that I want remove & return it
- .clear()
- .copy()
- .count(item)
- .index(item)

- Tuple (immutable)
  - Tuples = (2 , ‘c’, 7,6)
  - To convert tuple to list
    - Var = list(tubles_name)
  - Concatenate
    - +

```
tu = ("one" , "two" , 5  , "three")
a , b , _ , c = tu
print(a) #one
print(b) #two
print(c) #three
```

  -

- Set (is an unordered collection of items and unindexed)
  - Writing methods
    - Myset = {10 , 20 , 30 , 20 , 10}
  - Function
    - Myset.union(anotherSet , set , …) = myset | anotherSet
    - Myset.add(value)
    - .discard(value you wonna remove it without error)
    - Myset.update(Myset1) |=
      - بيدمجهم مع بعض بس مش بياخد المتكرر
    - Myset.difference(anotherSet) = myset – anotherset
      - اللى موجود فى الاولى ومش موجود فى التانيه
    - Myset.symmetric_difference (Myset1) ^
      - اللى مش موجوده فى الاتنين
    - Myset.intersection (Myset1) &
    - Myset.intersection_update(Myset1) &=
    - Myset.difference_update(Myset1) -=
      - هيعمل لناتج الديفيرينس ابديت ويحطها فى الاولى
    - Myset.symmetric_difference_update(Myset1) ^=
    - Set1.issuperset(set2)
      - set2 part of set1
    - set1.isupset(set2)

        - set1 part of set2
    - set1.disjoint(set2) (منفصلينْ)
- Dictionaries
  - Writing methods
    - grades = {
    - "arabic" : 50,
    - "English" : 70,
    - "german" : 90
    - }
    - Loop
      - for k in grades:
      - for x,y in dept.items():
      -
    - Print('german' in grades)
    - Print(grades.keys())
    - Print(grades.values())
    - Print(grades.items())
  - functions
    - Grades['islamic'] = 80
    - Del grades['English']
    - Del grades
    - Grades1 = grades2.copy()
    - Grades.pop[key]
    - Dic.popitem()
    - Grades.get('math' , 'not exist') return the value and if not exist return another msg
    - Grades1.update(grades2) grades1 + grades2
    - Dic.keys()
    - Dic.values()
    - Dic.items()
    - Dic.setdefault(key , value)

```
a = ("one" , "two", "three")
b = "value"
print(dict.fromkeys(a , b)) #{'one': 'value', 'two': 'value', 'three': 'value'}
```

    -
- map

❖ operations

- Arithmetic
  - =, +, -, *, **(exponent), /, //(floor divide), %
- Comparison
  - ==, !=, >, <, >=, <=
- Logical
  - And, or, not

- Bitwise
  - &, |, ^, ~, <<, >>
  - Ex
    - 10 -> 8 + 2
    - 75 -> 64 + 8 + 2 + 1
    - 10 & 75 = 10(8 + 2). بياخد المشترك
    - 10 | 75 = 75(64 + 8 + 2 + 1). كائنى بعمل اتحاد
    - 10 ^ 75 = 65(64 + 1) هياخد اللى مش مشترك
    - 10 << 3 = 80 (01010 000) بياخد 3 اصفار من اليمين ويحطهم فى الشمال
    - 10(1010) >> 3 = 1
      - 0000 1010
      - 0100 0001
    - 75 >> 3 = 9
      - 0100 1011
      - 0111 1001
- Assignment
- Membership
  - In, not in
    - If x in numbers:
    - If x not in numbers:
- Identity
  -

❖ Conditions
- If
- If … else
- Nested if
  - If … elif … elif … else
- Turnary

```
age = 18
age_user = int(input("enter your age : "))
print("Done, your age is valid " if age_user > age else
"Sorry, your age is not valid")
```

❖ Loops
- For
  - For I in “Egypt”:
  - For I,x in enumerate(“Egypt”)
  - For in range(start, condition, step)
    - For I in range(2,7,1)
  - format
    - num = 7
    - for i in range(1,11):
    - print("{}*{}={}" . format (i,num , i*num))
- While
-

- Function
  - Func with args
    - Def function_name (a,b): return ()
  - Fun with list
    - def sum(*nums): #tuble
    - result = 0
    - count = 0
    - for i in nums:
    - result = result + i
    - count = count + 1
    - return (result , count)
    -
    - result_sum = sum(2 , 4, 6, 3) #list
    - print(result_sum[0])
    - print(result_sum[1])
  - func with dictionary (kwargs)

```
def grades(**skills):
    print(type(skills)) #dict
    for language , precentage in skills.items():
        print(f"{language} => {precentage}")

# grades(html = "50%" , css = "60%" , js = "90%")
languages = {
    "html" :"50%",
    "css" : "60%",
    "js" : "90%"
}
grades(**languages)
```

- Lambda

```
hello = lambda name , age : f"the name is {name} & the age is {age}"
print(hello("meroo" , 20))
```

- Recursive

```
def main_word(word):
    if len(word) == 1:
        return word
    if word[0] == word[1]:
        return main_word(word[1:])
    return word[0] + main_word(word[1:])

print(main_word("wwworrlddd"))
```

- Module
  - Random

```
import random
print(random.randint(10 , 50))
```

```
import random as rn
print(rn.random()) # from 0 to 1
print(rn.randint(10 , 100)) # integer number from 10 to 100
a = random.randint(2 ,10, size = 6) #[4 9 6 8 2 6]

print(rn.uniform(10 , 100)) # decimal number from 10 to 100
a = random.uniform(2 ,10 , 5) # 5.24139675 5.84049289
3.83104411 8.26042491 5.57731417]
]
print(rn.randrange(100)) # integer number from 0 to 100
print(rn.randrange(0,100,2)) # integer number from 0 to 100
(odd)
print(rn.choice([4,6,7])) # choice random number
print(rn.sample(range(200) , 10) # choice 10 nums from 0 to
199
print(rn.shuffle(a) # items in list a will be shuffle
print(rn.uniform(2 , 10))   #3.5383602557941733
a = random.random((2 , 10))
print(a)   #[[0.64261358 0.55698388 0.59304565 0.16059278
0.36594248 0.76777121
0.04232465 0.81228508 0.15808882 0.4843274
[0.59338917 0.20505942 0.58565385 0.9934994  0.99986773
0.96498948
0.17749675 0.58416055 0.27800513 0.80648592]]
```

- import sys

```
import sys
print(sys.path)
sys.path.append(r"D:\Games")
print(sys.path)
```

- alias
  - import test as tt

❖ extern Package => pip

- in cmd
  - pip –version
  - pip list
  - to install package
    - pip install termcolor
    - pip install pyfiglet
      - termcolor & pyfiglet => are names of packages
    -
  - To upgrade the library
    - pip install pip –upgrade
- import
  - import pyfiglet
    - pyfiglet.figlet_format(“meroo”)
  - import datetime
    - print the current date & time
      - datetime.datetime.now()
      - datetime.datetime.now().year

        - datetime.datetime.now().month
        - datetime.datetime.now().day
        - datetime.datetime.now().time()
            - datetime.datetime.now().time().hour
            - datetime.datetime.now().time().minute
            - datetime.datetime.now().time().second
            - datetime.datetime.now().time().microsecond
            -
        -
        - datetime.datetime.min
        - datetime.datetime.max
        - datetime.datetime(year , month , day , hour , minute , second)
    - formate the date

```
print(my_date.strftime("%a - %b - %y")) #Sun - Jul - 03
print(my_date.strftime("%A - %B - %Y")) #Sunday - July - 2003
```

- import PIL
    - from PIL import Image

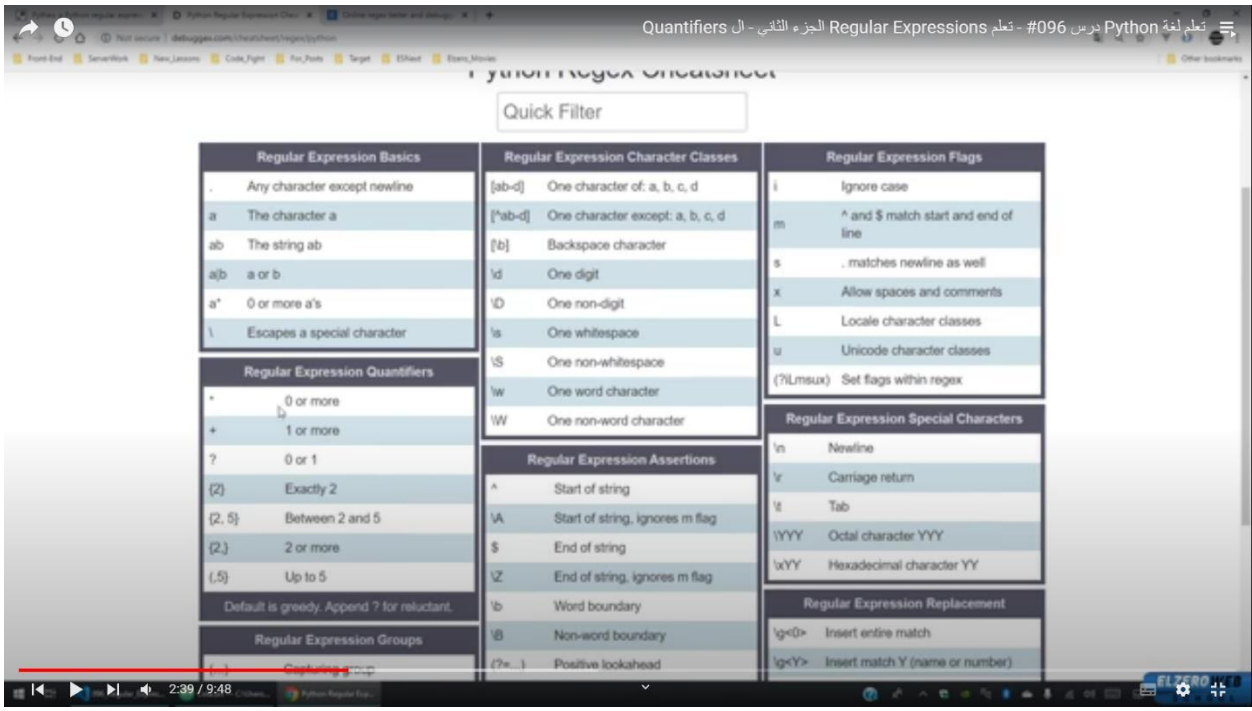
```
from PIL import Image
myImage = Image.open("C:\learnPython\game.jpg")
myImage.show()
box = (0,0,500,800)
crop_image = myImage.crop(box)
crop_image.show()
```

        - .crop(left , top , width , height)
- Import pylint => to improve the code
    - In terminal
        - Pylint.exe file_path

❖ Exception

- Raise exception(“”)
- Try: except: else: finally:
- Type of errors
    - ZeroDivisionError
    - NameError
    - ValueError
- Ex
    - Try:
        - Print(10 / 0)
        - Print(x)
        - Int(“Hello”)
    - Except ZeroDivisionError:
        - Print(“you can not divide by 0”)
    - Except NameError:
        - Print(“the value is not exist”)

        - Except ValueError:
            - Print(“you can not convert word hello to integer”)
        - Except:
            - Print(“exist error here”)

- Regural
    - Website
        - Pythex
        - Regx101.com
    - 

- Doc
    - ```
      def doc_func(name):
          """
          info about the function
           return name of person
           to explain documentation
          """
          return name
      print(doc_func.__doc__)
      #      info about the function
      #       return name of person
      #       to explain documentation
      ```
    -
- Iterable & iterator
    - Iterable => string, list , … except Number(Int & float)
    - Iter(iterable) = iterator
    - Next(iterator)
    - ```
      myString = "Meroo"
      myIterator = iter(myString)
      ```

```
print(next(myIterator)) #M
print(next(myIterator)) #e
print(next(myIterator)) #r
```

- generator

```
def myGenerator():
    yield 1
    yield 2
    yield 3
    yield 4
    yield 5

print(next(myGenerator())) #1
print(next(myGenerator())) #1
print("#" * 10) ##########
gen = myGenerator()
print(next(gen)) #1
print(next(gen)) #2
print("hello") #hello
print(next(gen)) #3
for g in gen:
    print(g, end=" ") # 4 5
```

- Decorator
  - No parameter

```
def myDecorator(func):
    def nestedFunc():
        print("before")
        func()
        print("after")
    return nestedFunc
@myDecorator
def sayHello():
    print("Hello")

sayHello()
```

  - with parameter

```
def myDecorator(func):
    def nestedFunc(num1 , num2):
        print("before")
        func(num1 , num2)
        print("after")
    return nestedFunc

@myDecorator
def sum(num1 , num2):
    print(num1 + num2)

sum(2 , 5)
```

- Zip

```
list = [1 , 2, 3, 4, 5]
list1 = ["A" , "B" , "C"]
```

```
print(zip(list , list1))    #<zip object at 0x000001DEC5A47E40>
for item in zip(list , list1):
    print(item)
# (1, 'A')
# (2, 'B')
# (3, 'C')
```

- file
  - mode
    - r -> read
    - a -> append
    - w -> rite
    - x -> create
  - paths
    - to know directory name

```
import os
print(os.path.dirname(os.path.abspath(__file__)))
#C:\learnPython\learn.py
```

    - to remove directory

```
import os
os.remove("moraa.txt")
```

    - absolute path => Starting is from root
      - if you wanna know absolute path

```
import os
print(os.path.abspath(__file__))  #
C:\learnPython\learn.py\main.py
```

    - relative path
      - to use it => you should know the Current Working Directory

```
import os
print(os.getcwd())  #C:\learnPython\learn.py
```

  - function
    - write
      - f = open('Desktop/python.txt' , 'w')
      - f.write("hello, my name is merOoo")
      - f.close()
      - f.writelines(list_name)
    - read
      - f = open('Desktop/python.txt' , 'r')
      - f.read()
      - f.readline()
      - f.readlines()
      - f.seek(5) => skept 5 characters then print

```
myFile = open("moraa.txt" , "r")
myFile.seek(5)
print(myFile.readline())
```

- append
  - f = open('Desktop/python.txt' , 'a')
  - f.write(' and my age is 20 years')
  - f.writelines(myList)
  - f.truncate(5) => cut the text

```
myFile = open("moraa.txt" , "a")
print(myFile.truncate(5)) #5
print(open("moraa.txt" , "r").readline())
```

  - f.tell() => place of curser
    - calc the new line with 2byte => (\r\n)
    - Return number of characters
  - f.close()
- build-in-functions
  - name & mode & encoding

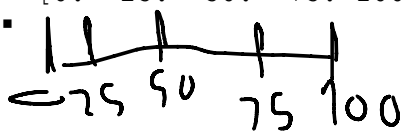
```
myFile = open("moraa.txt" , "r")
print(myFile)
#C:\learnPython\learn.py\main.py <_io.TextIOWrapper
name='moraa.txt' mode='r' encoding='cp1252'>
print(myFile.name) #moraa.txt
print(myFile.mode) #r
print(myFile.encoding) #cp1252
```

❖ Statistics
- Import statistics as st
- A = st.Mean([2 , 6 , …])
- A = St.harmonic_mean([2 6 , …])
- A = st.median([2,6,…])
- A = st.media_low([4,5,6,7]) #5
- A = st.media_high([4,5,6,7]) #5
- A = st.mode([2,3,2]) #2
- Standard deviation
  - A = st.stdev([3.2,6,…])
- A = st.variance([6,7,…])

❖ Numpy
- import numpy as np
  - .ipynb_checkpoints\numpy-checkpoint.ipynb
  - SIN
    - import math as ma
    - a = ma.sin(30 * (np.pi/180)) #or
      - a = ma.sin(np.deg2rad(30)) #or
      - a = ma.sin(ma.radians(30))

- print(a) #0.49999999999999994

- print(np.mod(21 , 5)) #1
- METRICS
  - Array & shape
    - a = np.array([[1,2,3],[4,5,6]])
    - print(a)
      - [[1 2 3]
      - [4 5 6]]
    - Print(a.shape)
      - (2 , 3)
  - With range
    - `a = np.array([range(i , i+4) for i in [2,4,6]])`
    - `print(a)`
    - `[[2 3 4 5]`
    - `[4 5 6 7]`
    - `[6 7 8 9]]`
  - Empty
    - A = empty((2,4))
      - `[[0.00e+000 0.00e+000 0.00e+000 0.00e+000]`
      - `[0.00e+000 7.35e-321 0.00e+000 0.00e+000]]`
  - Zeros, Ones, Identical, diagonal & Full
    - A = ones((4 , 6))
    - A = zeros((2 , 5))
    - A = eye(6)
    - a = diag(array([1 ,3 , 5, 7, 9]))
      - `[[1 0 0 0 0]`
      - `[0 3 0 0 0]`
      - `[0 0 5 0 0]`
      - `[0 0 0 7 0]`
      - `[0 0 0 0 9]]`
    - a = diag(array([1 ,3 , 5, 7, 9]) , k = 2)
      - `[[0 0 1 0 0 0 0]`
      - `[0 0 0 3 0 0 0]`
      - `[0 0 0 0 5 0 0]`
      - `[0 0 0 0 0 7 0]`
      - `[0 0 0 0 0 0 9]`
      - `[0 0 0 0 0 0 0]`
      - `[0 0 0 0 0 0 0]]`
      - NOTE -> it will increase by 2 cols & 2 rows
    - A = full((3 , 5) , 20)
  - Arrange & reshape
    - A = arange(10)

        - [0 1 2 3 4 5 6 7 8 9]
    - A = arrange(10 , 20)
        - [10 11 12 13 14 15 16 17 18 19]
    - A = arrange(10 , 20 , 2)
        - [10 12 14 16 18]
    - a = arange(5 , 25).reshape(5, 4)
        - [[5 6 7 8]
        - [9 10 11 12]
        - [13 14 15 16]
        - [17 18 19 20]
        - [21 22 23 24]]
        - NOTE -> from 5 to 25 is 20 numbers so that we need matricx is rows * cols = 20
    - A = arange(27).reshape(3,3,3)
        - [[[0 1 2]
        - [3 4 5]
        - [6 7 8]]
        -
        - [[9 10 11]
        - [12 13 14]
        - [15 16 17]]
        -
        - [[18 19 20]
        - [21 22 23]
        - [24 25 26]]]
        -
    - c = linspace(0 , 100 , 5)
        - [0. 25. 50. 75. 100.]
        - 

- pandas
    - Pandas.ipynb
- OOP
    - Magic Methods
        - Class
            - Object_name = class_name()
            - Object_name.__class__ => return class_name

```
class learn:
    def __init__(self):
        self.list = ["css" , "html" , "js" , "react"]
    def __str__(self):
        return f"{self.list}"
    def __len__(self):
```

```
        return len(self.list)
test = learn()
print(test) #['css', 'html', 'js', 'react']
print(len(test))    #4
```

- @classmethod & @staticmethod

```
class Member:
    counter = 0
    def __init__(self , first_name , last_name):
        self.fname = first_name
        self.lname = last_name
    def info_person(self):
        return f"Hello {self.fname} {self.lname}"
    @classmethod
    def number_users(cls):
        cls.counter += 1
    @staticmethod
    def number_users_with_static():
        Member.counter +=1

person1 = Member("amira" , "hassan")
print(person1.info_person())
print(Member.number_users())
print(person1.counter)
print(person1.number_users())
```

- Inheritance

```
class Food:
    def __init__(self , name):
        self.fname = name
        print(f"{self.fname} is from Parent Class")

class Apple(Food):
    def __init__(self , name):
        super().__init__(name)
        print(f"{self.fname} is from derived class")

a1 = Apple("Yellow")
print(a1.fname)

# Yellow is from Parent Class
# Yellow is from derived class
# Yellow
```

- Projects
  - Calc average of subjects
    - math = int(input ("math : "))
    - science = int(input("science : "))
    - computer = int(input("computer : "))
    - total = math + science + computer
    - avg = total / 3
    - print(avg)
- Escape Sequences Characters
  - \b => Back Space
  - \n => new line
  - \r => carriage return
  - \t => Horizontal Tab
  - \xhh => Character Hex Value
    - Print("\x4F\x73") # Os
- Concatenate
  - +
- Notes
  - Iterable
    - String , list , …. Except Integer & Float
  - The variables will have the same id if they have the same values -> because they have the same reference

```
x = 5
y = 5
print('x : ',id(x))    #x : 140732599473064
print('y : ' , id(y))  #y : 140732599473064

y = 8
print('\nx : ',id(x))   #x : 140732599473064
print('y : ' , id(y))        #y : 140722110391304

print('\nx : ',x)    #5
print('y : ' , y)    #8
```

  - Run Time Error

```
x = 5
def ff(d):
    x = x + 1
    print(x)
ff(x)
```

  - Tuples

```
t = ('c',)
s = ('c')

print('t : ',type(t))        #t :  <class 'tuple'>
print('s : ',type(s))        #s :  <class 'str'>
```

```
i = 5
t = 5,
```

```
print('t : ',type(i))    #t :  <class 'int'>
print('s : ', type(t))     #s :  <class 'tuple'>
```

- swap

```
x = 4
y = 6
(x,y) = (y,x)
print('x : ' , x)    # 6
print('y : ' , y)    # 4
```

- Alias

```
l = [1 , 3, 4]
l1 = [2 , 6 , 7]
l = l1
l[0] = 1000
print(l1)    # [1000, 6, 7]
```

- Cloning a List

```
l = [1 , 3, 4]
l1 = [2 , 6 , 7]
l = l1[:]
l[0] = 1000
print(l1)    # [2, 6, 7]
```

- remove_dups

```
def remove_dups(l1 , l2):
    l1_copy = l1[:]
    for i in l1_copy:
        if i in l2:
            l1.remove(i)

l1 = [1 , 2, 3, 5]
l2 = [2 , 3]
remove_dups(l1 , l2)
print(l1)
```

- all data in python is Object
- a, b, c = 7 , 3 , 5
- to concatenate number with string

```
age = 20
name = "meroo"
print("my name is %s and my age is %d" %(name , age))
print("my name is {} and my age is {}".format(name , age))
num = 35.5720
name = "meroo hassan"
print("my name is {:.5s} and the num is {:.2f}".format(name
, num))
num = 243476985758790
print("the num is {:_d}".format(num)) #the num is
243_476_985_758_790

a , b , c = "one" , "two" , "three"
print("hello {1} {0} {2}".format(a , b , c)) #hello two one
three
```

▪

< Questions >

HackerRank

- Is_leapYear

```
def is_leap(year):
    leap = False
    if year%4==0 and year%400==0 and year%100!=0:
        leap = True
    return leap

year = int(input())
print(is_leap(year))
```

- print_numbers_without_space

```
def loop(n):
    str_num = ""
    for i in range(1, n + 1):
        str_num = str_num + str(i)
    return str_num

n = int(input())
print(loop(n))
```

SCIKIT-LEARN SYNTAX